

Express Mail No. EL750740750US

Docket No. 00-117-DSK

SELF-DEFINING DATA UNITS**CROSS REFERENCE TO PROVISIONAL AND RELATED APPLICATIONS**

This application claims the benefit of the filing date of corresponding U.S. Provisional Patent Application No. _____, entitled "System for providing a policy-based demand and use of functions like virtual volumes, instant copy, RAID, etc.", filed June 19, 2000. In addition, the present invention is related to applications entitled A SYSTEM TO SUPPORT DYNAMICALLY FLEXIBLE DATA DEFINITIONS AND STORAGE REQUIREMENTS, serial no. _____, Attorney Docket Number 00-059-DSK; EFFECTING INSTANT COPIES IN A DYNAMICALLY MAPPED SYSTEM, serial no. _____, Attorney Docket Number 00-060-DSK; USING CURRENT RECOVERY MECHANISMS TO IMPLEMENT DYNAMIC MAPPING OPERATIONS, serial no. _____, Attorney Docket Number 00-061-DSK; DYNAMICALLY CHANGEABLE VIRTUAL MAPPING SCHEME, serial no. _____, Attorney Docket Number 00-062-DSK; FLOATING VIRTUALIZATION LAYERS, serial no. _____, Attorney Docket Number 00-116-DSK, and RECOVERY OF DYNAMIC MAPS AND DATA MANAGED THEREBY, serial no. _____, Attorney Docket Number 00-063-DSK, which are filed even date hereof, assigned to the same assignee, and incorporated herein by reference.

Docket # 00-117-DSK

Docket No. 00-117-DSK

BACKGROUND OF THE INVENTION**1. Technical Field:**

The present invention relates to an improved data processing system and, in particular, to recovery of virtualization structures. Still more particularly, the present invention provides a method and apparatus for providing self-defining data units.

2. Description of Related Art:

Maps are used in a disk controller to convert a host based Logical Unit (LUN) and Logical Block Address (LBA) to a controller based LUN and LBA. A mapping system is necessary for a disk controller to provide features such as virtual volumes, data compression, and snapshot. In fact, maps are used in current controller designs to facilitate the use of Redundant Array of Independent Disk (RAID) devices.

A problem that arises when using a mapped based architecture is where to store the maps. Current map designs use anywhere from four megabytes for a very simple map to dynamic mapping systems that use twelve megabytes or more. As the sizes of disks increase and the sizes of system configurations increase, it is not inconceivable that these systems will require maps that are several gigabytes in size.

These large structures make recovery of the data, in the case of a lost or failed virtual map, a time-consuming and complicated process. In addition, some

Thus, it would be advantageous to provide self-defining data and mapping elements.

Docket No. 00-117-DSK

SUMMARY OF THE INVENTION

The present invention provides a mechanism for storing self-defining data and mapping elements with either a fixed set of allowed structures or types or with the structures and types determined by rules. Recovery is enhanced by the use of backward and forward pointers between data and mapping elements for the data elements in the order written by the management algorithm. Recovery is also enhanced by the use of companion pointers with metadata. The companion pointers may include pointers to data or mapping elements that are part of the same structural grouping. For example these pointers may point to the elements that make up a redundancy stripe or the elements that make up a mapping sub-tree. The metadata may describe the structural grouping. The metadata may also include pointers to the previous and/or next versions of the same elements. For example, the metadata may include a pointer to the previous older version of a data block or to the location where the next version of the data block will be stored.

005144-13903

Docket No. 00-117-DSK

BRIEF DESCRIPTION OF THE DRAWINGS

The novel features believed characteristic of the invention are set forth in the appended claims. The invention itself, however, as well as a preferred mode of use, further objectives and advantages thereof, will best be understood by reference to the following detailed description of an illustrative embodiment when read in conjunction with the accompanying drawings, wherein:

Figure 1 depicts a pictorial representation of a distributed data processing system in which the present invention may be implemented;

Figure 2 is a block diagram of a storage subsystem in accordance with a preferred embodiment of the present invention;

Figure 3 is a block diagram of a data structure in accordance with a preferred embodiment of the present invention; and

Figure 4 is an example of a metadata data structure in accordance with a preferred embodiment of the present invention.

Docket No. 00-117-DSK

Docket No. 00-117-DSK

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

With reference now to the figures, **Figure 1** depicts a pictorial representation of a distributed data processing system in which the present invention may be implemented. Distributed data processing system **100** is a network of computers in which the present invention may be implemented. Distributed data processing system **100** contains a network **102**, which is the medium used to provide communications links between various devices and computers connected together within distributed data processing system **100**. Network **102** may include permanent connections, such as wire or fiber optic cables, or temporary connections made through telephone connections.

In the depicted example, a server **104** is connected to network **102** along with storage subsystem **106**. In addition, clients **108**, **110**, and **112** also are connected to network **102**. These clients **108**, **110**, and **112** may be, for example, personal computers or network computers. For purposes of this application, a network computer is any computer, coupled to a network, which receives a program or other application from another computer coupled to the network. In the depicted example, server **104** provides data, such as boot files, operating system images, and applications to clients **108-112**. Clients **108**, **110**, and **112** are clients to server **104**. Distributed data processing system **100** may include additional servers, clients, and other devices not shown. Distributed data processing system **100** may be implemented as one or more of

09254641-1399000

Docket No. 00-117-DSK

a number of different types of networks, such as, for example, an intranet, a local area network (LAN), or a wide area network (WAN). Network 102 contains various links, such as, for example, fiber optic links, packet switched communication links, enterprise systems connection (ESCON) fibers, small computer system interface (SCSI) cable, wireless communication links. In these examples, storage subsystem 106 may be connected to server 104 using ESCON fibers. **Figure 1** is intended as an example and not as an architectural limitation for the present invention.

Turning next to **Figure 2**, a block diagram of a storage subsystem is depicted in accordance with a preferred embodiment of the present invention. Storage subsystem 200 may be used to implement storage subsystem 106 in **Figure 1**. As illustrated in **Figure 2**, storage subsystem 200 includes storage devices 202, interface 204, interface 206, cache memory 208, processors 210-224, and shared memory 226.

Interfaces 204 and 206 in storage subsystem 200 provide a communication gateway through which communication between a data processing system and storage subsystem 200 may occur. In this example, interfaces 204 and 206 may be implemented using a number of different mechanisms, such as ESCON cards, SCSI cards, fiber channel interfaces, modems, network interfaces, or a network hub. Although the depicted example illustrates the use of two interface units, any number of interface cards may be used depending on the implementation.

00-117-DSK

Docket No. 00-117-DSK

In this example, storage subsystem 200 is a shared virtual array. Storage subsystem 200 is a virtual storage system in that each physical storage device in storage subsystem 200 may be represented to a data processing system, such as client 104 in **Figure 1**, as a number of virtual devices. In this example, storage devices 202 are a set of disk drives set up as a redundant array of independent disks (RAID) system. Of course, other storage devices may be used other than disk drives. For example, optical drives may be used within storage devices 202. Further, a mixture of different device types may be used, such as, disk drives and tape drives.

Data being transferred between interfaces 204 and 206 and storage devices 202 are temporarily placed into cache memory 208. Additionally, cache memory 208 may be accessed by processors 210-224, which are used to handle reading and writing data for storage devices 202. Shared memory 226 is used by processors 210-224 to handle and track the reading and writing of data to storage devices 202. In particular, processors 210-224 are used to execute instructions for routines used in snapshot copy operations.

The present invention manages virtual storage facilities comprising an organization of computer equipment, for example, a host network, data transfer means, storage controller means and permanent storage means and attachment means connecting these devices together. The data storage facilities also may include

Docket No. 00-117-DSK

management information associated with data units such that the management information provides an inventory of capabilities with upper and lower boundaries that may limit the options available to store the data and still meets a user's criteria. All RAID stripe groups may be read at once up to the point of anticipated performance requirements. If all RAID stripe groups are read at once, but the system does not meet a newly imposed performance requirement then the data may be rewritten to a higher performance capability. Such management information may be independent of attributes of characteristics of the elements of the physical storage subsystem actually used to store the data objects, but may consist of imputed associations with those attributes through, for example, changeable rule sets, processes or algorithms. These rule sets, processes or algorithms may be changed by user demand or via processes that may monitor data object usage and manipulation. The storage of data objects may be adjusted to comply with modifications in the rules sets, processes or algorithms, for example.

With reference to **Figure 3**, a block diagram of a data structure is illustrated in accordance with a preferred embodiment of the present invention. Data structure **300** includes data elements D1 **301**, D2 **302**, D3 **303**, D4 **304**, D5 **305**, D6 **306**, and D7 **307**. Each data element includes metadata uniquely associated with the data such that installation management criteria, boundaries, and preferences for each data unit and

Docket No. 00-117-DSK

attributes for the data units are maintained. This metadata may include time sequencing of metadata (time stamp), location of stored data, structure definition pointers including size parameters, pointers to related metadata units, management rules, sequencing rules, and management functions invoked to accomplish management rules.

The management rules may include performance criteria, reliability criteria, availability criteria, and capacity criteria. The sequencing rules may include logical rules, time rules, and structure rules. Management functions may include RAID, parity, multiple parity, and other known functions that may be invoked to accomplish management rules. Management rules, sequencing rules, and management functions may also be stored in the metadata as pointers to the rules or functions.

Furthermore, each data element may include pointers to the next or previous version in a time sequence. For example, data element D1 301 includes a pointer to the next version of updated data, D2 302. Consequently, data element D2 includes a pointer to the previous version, D1. Each data element may include pointers to the next or previous data element in a logical sequence, such as a next track in a sequence. For example, data element D3 303 may include a pointer to D4 304 as the next track and D4 may include a pointer to D5 as the next track. Conversely, D5 may include a pointer to D4 as the

Docket No. 00-117-DSK

previous data element in the logical sequence and D4 may include a pointer to D3 as the previous data element.

Data elements D2 **302** and D4 **304** may include metadata to indicate that they are mirrored with pointers to the mirrored copies. Therefore, one can get twice the read performance and improved availability. Data elements D5, D6, and D7 may include metadata to indicate that they are part of a RAID stripe and the available read bandwidth is three drives.

The metadata may be stored separate from the data. Thus, each data element may include a virtual address (VA) pointing to the host view of the stored data. For example, D1 **301** includes VA **311**, D2 **302** includes VA **312**, D3 **303** includes VA **313**, D4 **304** includes VA **314**, D5 **305** includes VA **315**, D6 **306** includes VA **316**, and D7 **307** includes VA **317**.

The data elements in **Figure 3** may be mapping elements. Mapping elements may include forward and backward pointers to mapping elements. If the mapping tables are lost or corrupted, then the mapping may be recovered by finding one or more of the data elements, rebuilding the mapping by following the all the links to the other data elements, and reestablishing the mapping entries with the virtual address stored in the data element.

Figure 4 is an example of a metadata data structure in accordance with a preferred embodiment of the present invention. Such a data structure **400** may include all or

Docket No. 00-117-DSK

a subset of, but is not limited to, the items described here.

Metadata item **405** is/contains an identifier to uniquely identify the data element (or elements) described by this metadata structure. Metadata item **410** identifies the data unit or units associated with this data element (or elements). This may be, for example, a virtual data unit address, or a pointer to such an address or an input to an algorithm to calculate an address. This item **410** is used during virtual mapping system recovery to rebuild the map.

Metadata item **412** contains a pointer to the current physical storage location wherein this data element (or elements) is stored. Item **412** may alternatively contain an input to an algorithm used to calculate the physical storage location pointer or address. Item **412** may alternatively contain an indicator that this data element is not stored anywhere, but is un-allocated or assumed to have a zero (or some other default) data content. Item **412** may also point to a list of multiple physical locations where duplicate copies of this data element are stored.

Metadata item **414** contains a history of one or more previous physical storage locations. If the current physical storage location is not accessible, it might be possible to recover the data from a previously used location.

Metadata item **416** contains an indication of the type of virtual mapping structure used to map from the

005464-13500

Docket No. 00-117-DSK

associated data unit or units to this data element or elements. The types may include but are not limited to, an algorithmic mapping, a multi-level mapping tree, and a single level or two level mapping table.

Metadata item **418** contains pointers to related data elements. There may be multiple such pointers. The pointers may be to the physical location of the data itself, or to the associated metadata, or may be an input to an algorithm used to calculate the location of the related data element. A data element may be related to this data element in one or more of a variety of ways, for example; a data element may be another element in a redundancy group, or the next or previous element written on a log structured file or it may be a parent or child data element in a tree structure or the previous or next element in a tree sequence, etc.

Metadata item **420** contains a pointer to a list of related data elements. This allows a single list of related data elements to be used by all the data elements in one grouping.

Metadata **422** contains one or more pointers to allow access to related data units. These may be, for example, data unit virtual addresses or inputs to an algorithm to calculate such addresses. Related data units may be, for example, previous or next data units in some virtual address sequence, or data units involved in a point-in-time copy operation, etc.

These pointers to related data elements (metadata items **418** and **420**) and to related data units (metadata

0054341 1 0000

Docket No. 00-117-DSK

items **422**) along with other metadata items such as the rules (items **426**, **428**, **430**, etc.) allow improved recovery of virtual mapping systems. Pointers to related data elements may allow fast searches during map recovery by following the pointers of a specific type, such as the pointers to previous and companion data elements in a log structured file. Another example would be to use the metadata items **422** to search for the data elements associated with a particular data unit.

Metadata item **424** contains one or more timestamps indicating the time or relative sequence number of some event or events associated with this data element. This may include, for example, the time of the last change to the data, the time of the last update to this metadata, the time this data element was last moved, etc. The timestamp(s) in metadata item **424** may be used to enable recovery of mapping structures by, for example, distinguishing which of multiple data elements is the most current version, or which of multiple data elements contains the version of the data closest to the desired recovery checkpoint, etc.

Metadata item **426** contains rules for processing this data element and possibly related data elements. These rules may include, for example, an indication of the specific data format or encoding of this data element, or an algorithm for interpreting the data element format, or the RAID level of an associated redundancy group, thus indicating how the data elements in the redundancy group should be processed.

Metadata item **430** contains rules for the order in which data elements should be processed. This may include, for example, rules to indicate that the data elements of a certain set must be processed in timestamp order, oldest first, or that the data elements in a redundancy group must all be processed before the data elements in the next redundancy group or that the data elements that are pointed to as "previous" in a logical sequence should be processed first.

Metadata item **432** contains data unit attributes for the data unit or units associated with this data element. The may, include for example:

Performance criteria (item 434):

- a) sustainable data transfer rate
- b) sustainable SIO/sec
- c) parallel SIO

Availability criteria (item 436):

- a) time to first accessibility of data
- b) time to hold off new users for consistency checks

Reliability criteria (item 438):

- a) allowed probability of data block loss
- b) allowed probability of data file loss

Capacity Management criteria (item 440):

- a) maximum size of data unit

Docket No. 00-117-DSK

In order to facilitate the processing of the data elements, the metadata structure 400 may also include pointers to executable functions for interpreting and executing management rules (item 442). The data elements can then have specific associated specialized functions, and processing of the data elements can proceed quickly without having to determine by logic which such functions are needed to process each data element.

The metadata structure may also contain boundary information for this data element (item 444) or for related data units (item 446). This boundary information may include, for example, the current size, or the current address range covered by the data element or data unit, and/or the maximum size allowed for this data element and/or data unit, and/or the range of allowed locations to store this data element, etc.

The access history for this data element and/or related data elements or data units may also be included, as in items 448 and 450. This may include information such as, for example, the count of accesses or frequency of accesses to this or related data elements or data units, and/or an indicator of the priority of the accesses, etc. This access history (448 and/or 450) may then be used to prioritize and order recovery processing of the data elements.

It is important to note that while the present invention has been described in the context of a fully functioning data processing system, those of ordinary skill in the art will appreciate that the processes of

Docket No. 00-117-DSK

the present invention are capable of being distributed in the form of a computer readable medium of instructions and a variety of forms and that the present invention applies equally regardless of the particular type of signal bearing media actually used to carry out the distribution. Examples of computer readable media include recordable-type media such a floppy disc, a hard disk drive, a RAM, and CD-ROMs and transmission-type media such as digital and analog communications links.

The description of the present invention has been presented for purposes of illustration and description, but is not intended to be exhaustive or limited to the invention in the form disclosed. Many modifications and variations will be apparent to those of ordinary skill in the art. The embodiment was chosen and described in order to best explain the principles of the invention, the practical application, and to enable others of ordinary skill in the art to understand the invention for various embodiments with various modifications as are suited to the particular use contemplated.

005117-1-1390